```
                                                       M 1








BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSSS  RRRRRRRRRRR      TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSSS  RRRRRRRRRRR      TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSSS  RRRRRRRRRRR      TTTTTTTTTTTTTTT  LLL
BBB         BBB  AAA         AAA  SSS              RRR         RRR        TTT        LLL
BBB         BBB  AAA         AAA  SSS              RRR         RRR        TTT        LLL
BBB         BBB  AAA         AAA  SSS              RRR         RRR        TTT        LLL
BBB         BBB  AAA         AAA  SSS              RRR         RRR        TTT        LLL
BBB         BBB  AAA         AAA  SSS              RRR         RRR        TTT        LLL
BBB         BBB  AAA         AAA  SSS              RRR         RRR        TTT        LLL
BBBBBBBBBBBB     AAA         AAA     SSSSSSSSS     RRRRRRRRRRR            TTT        LLL
BBBBBBBBBBBB     AAA         AAA     SSSSSSSSS     RRRRRRRRRRR            TTT        LLL
BBBBBBBBBBBB     AAA         AAA     SSSSSSSSS     RRRRRRRRRRR            TTT        LLL
BBB         BBB  AAAAAAAAAAAAAAA              SSS  RRR   RRR              TTT        LLL
BBB         BBB  AAAAAAAAAAAAAAA              SSS  RRR   RRR              TTT        LLL
BBB         BBB  AAAAAAAAAAAAAAA              SSS  RRR   RRR              TTT        LLL
BBB         BBB  AAA         AAA              SSS  RRR      RRR           TTT        LLL
BBB         BBB  AAA         AAA              SSS  RRR      RRR           TTT        LLL
BBB         BBB  AAA         AAA              SSS  RRR      RRR           TTT        LLL
BBBBBBBBBBBB     AAA         AAA  SSSSSSSSSSSS     RRR         RRR        TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB     AAA         AAA  SSSSSSSSSSSS     RRR         RRR        TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB     AAA         AAA  SSSSSSSSSSSS     RRR         RRR        TTT        LLLLLLLLLLLLLLL
```

```
**FILE**ID**BPAMESAG




BBBBBBBB    PPPPPPPP      AAAAAA    MM      MM  EEEEEEEEEE    SSSSSSSS    AAAAAA      GGGGGGGG
BBBBBBBB    PPPPPPPP      AAAAAA    MM      MM  EEEEEEEEEE    SSSSSSSS    AAAAAA      GGGGGGGG
BB      BB  PP      PP  AA      AA  MMMM  MMMM  EE          SS          AA      AA  GG
BB      BB  PP      PP  AA      AA  MMMM  MMMM  EE          SS          AA      AA  GG
BB      BB  PP      PP  AA      AA  MM  MM  MM  EE          SS          AA      AA  GG
BB      BB  PP      PP  AA      AA  MM  MM  MM  EE          SS          AA      AA  GG
BBBBBBBB    PPPPPPPP    AA      AA  MM      MM  EEEEEEEE      SSSSSS    AA      AA  GG
BBBBBBBB    PPPPPPPP    AA      AA  MM      MM  EEEEEEEE      SSSSSS    AA      AA  GG
BB      BB  PP          AAAAAAAAAA  MM      MM  EE                  SS  AAAAAAAAAA  GG  GGGGGG
BB      BB  PP          AAAAAAAAAA  MM      MM  EE                  SS  AAAAAAAAAA  GG  GGGGGG
BB      BB  PP          AA      AA  MM      MM  EE                  SS  AA      AA  GG      GG      ....
BB      BB  PP          AA      AA  MM      MM  EE                  SS  AA      AA  GG      GG      ....
BBBBBBBB    PP          AA      AA  MM      MM  EEEEEEEEEE  SSSSSSSS    AA      AA    GGGGGG        ....
BBBBBBBB    PP          AA      AA  MM      MM  EEEEEEEEEE  SSSSSSSS    AA      AA    GGGGGG        ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
0001  0 !
0002  0 !<blf/width:80>
0003  0 !
0004  0 MODULE bpa$mesag (                                  ! Message send/receive
0005  0                 IDENT = '1-272'                     ! File: BPAMESAG.B32 EDIT:MDL1272
0006  0                 ) =
0007  1 BEGIN
0008  1 !
0009  1 !****************************************************************************
0010  1 !*                                                                          *
0011  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0012  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0013  1 !*  ALL RIGHTS RESERVED.                                                    *
0014  1 !*                                                                          *
0015  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0016  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0017  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0018  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0019  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0020  1 !*  TRANSFERRED.                                                            *
0021  1 !*                                                                          *
0022  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0023  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0024  1 !*  CORPORATION.                                                            *
0025  1 !*                                                                          *
0026  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0027  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0028  1 !*                                                                          *
0029  1 !*                                                                          *
0030  1 !****************************************************************************
0031  1 !
0032  1 !<blf/uppercase_key>
0033  1 !<blf/lowercase_user>
0034  1 !
0035  1
0036  1 !++
0037  1 ! FACILITY: BASIC-PLUS AME
0038  1 !
0039  1 ! ABSTRACT:
0040  1 !
0041  1 !       This module simulates some subfunctions of the RSTS monitor directive
0042  1 !       .MESAG.
0043  1 !       bpa$mess_dcl corresponds to SR$DCL
0044  1 !       bpa$mess_lcl     - " -     to SR$LCL
0045  1 !       bpa$mess_rcv     - " -     to SR$RCV
0046  1 !       bpa$mess_rem     - " -     to SR$REM
0047  1 !
0048  1 ! ENVIRONMENT: Native mode VAX processor, User mode.
0049  1 !
0050  1 ! AUTHOR: V.Eriksson (CAP), CREATION DATE: 10 - May - 1979
0051  1 !
0052  1 ! MODIFIED BY:
0053  1 !
0054  1 !       VERSION X01
0055  1 !
0056  1 !       V.ERIKSSON (CAP), 27-JUNE-79
0057  1 ! 260   - The definition of IOSB (status block returned by QIOW) was wrong
```

```
;     58         0058  1 !          and has been changed.
;     59         0059  1 !
;     60         0060  1 !        V.ERIKSSON (CAP), 27-JUNE-79
;     61         0061  1 ! 261    - A firqb-subfunction value has been inserted in bpa$mess_rcv.
;     62         0062  1 ! 1-262 - Remove REQUIRE 'REQ:AME' and added necessary
;     63         0063  1 !           BPA prefixes on the names.  JBS 02-OCT-1979
;     64         0064  1 ! 1-263 - Remove the XRB.  Instead, the caller will pass the necessary
;     65         0065  1 !           fields.  This is needed because in the VAX-11 BASIC environment
;     66         0066  1 !           addresses are 32-bits long, and the XRB only has 16 bits for
;     67         0067  1 !           the buffer address.  JBS 04-OCT-1979
;     68         0068  1 ! 1-264 - Add an exit handler, so that the logical name will be deleted
;     69         0069  1 !           when the image exits, in case it is unable to delete it.
;     70         0070  1 !           JBS 05-OCT-1979
;     71         0071  1 ! 1-265 - Correct some comments describing receiver selection
;     72         0072  1 !           and return 0 for job number.  JBS 07-OCT-1979
;     73         0073  1 ! 1-266 - Remember the parameter string, so it can be returned with each
;     74         0074  1 !           message segment read.  JBS 07-OCT-1979
;     75         0075  1 ! 1-267 - Correct an error in a comment.  JBS 16-OCT-1979
;     76         0076  1 ! 1-268 - If the permanent mailbox is already created and we do not have
;     77         0077  1 !         SYSNAM priv. to put the name of the mailbox in the table then
;     78         0078  1 !         instead of just signalling, first delete the permanent mailbox.
;     79         0079  1 !         FM 1-OCT-80
;     80         0080  1 ! 1-269 - Do not interpret FIRQB[FQB$W_BMAX] to be anything other than
;     81         0081  1 !         negative or positive.  Negative or zero means use temporary mailbox,
;     82         0082  1 !         and positive means use permanent mailbox. Use a constant for the
;     83         0083  1 !         size of message, when creating the mailbox. The buffer quota for
;     84         0084  1 !         the mailbox should be the system default. When these changes were
;     85         0085  1 !         made, BAS$SYS was also modified to support small send/receive.
;     86         0086  1 !         This call is simulated to look like a large send/receive to this
;     87         0087  1 !         module. As a result of adding small send/receive, we will now
;     88         0088  1 !         signal illegal sys usage instead of account or device in use
;     89         0089  1 !         if this is a small send/receive to be compatible with RSTS. FM 21-FEB_81.
;     90         0090  1 ! 1-270 - Use LIB$GET_EF to allocate event flags for $QIOWs.  PLL 30-Nov-81
;     91         0091  1 ! 1-271 - Deallocate an active ISB/LUB/RAB after trying to read a mailbox
;     92         0092  1 !         message and not receiving one.  MDL 11-Oct-1982
;     93         0093  1 ! 1-272 - fix bug introduced in previous edit; can't assume we have a valid
;     94         0094  1 !         ISB/LUB/RAB all the time.  MDL 26-Apr-1983
;     95         0095  1 !--
;     96         0096  1
;     97         0097  1 !
;     98         0098  1 !<blf/page>
;  100        0099  1 !
;  101        0100  1 ! SWITCHES:
;  102        0101  1 !
;  103        0102  1
;  104        0103  1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
;  105        0104  1
;  106        0105  1 !
;  107        0106  1 ! TABLE OF CONTENTS:
;  108        0107  1 !
;  109        0108  1
;  110        0109  1 FORWARD ROUTINE
;  111        0110  1     bpa$mesag,                          ! Main routine, calls the others
;  112        0111  1     bpa$mess_dcl,                               ! Simulates declare receiver
;  113        0112  1     mess_rcvname : NOVALUE,     ! Called by bpa$mess_decl and bpa$mess_lcl to
;  114        0113  1     ! create a logical mailbox name
;  115        0114  1     bpa$mess_lcl,                       ! Simulates send local data message
;  116        0115  1     bpa$mess_send : NOVALUE,    ! Called by bpa$mess_lcl to send a message
;  117        0116  1     bpa$mess_rcv,                       ! Simulates receive local data message
;  118        0117  1     read_mailbox,               ! Called by bpa$mess_rcv to read a mailbox
;  119        0118  1     ! into a buffer
;  120        0119  1     bpa$mess_rem,                               ! Simulates remove receiver
;  121        0120  1     bpa$mess_clear,             ! Called by bpa$mess_rem to perform remove
;  122        0121  1     exit_handler : NOVALUE;                     ! Called at image exit
;  123        0122  1
;  124        0123  1 ! receiver
;  125        0124  1 !
;  126        0125  1 ! INCLUDE FILES:
;  127        0126  1 !
;  128        0127  1
;  129        0128  1 REQUIRE 'RTLIN:BPASTRUCT';
;  130        0219  1
;  131        0220  1 REQUIRE 'RTLIN:BPAFQBDEF';
;  132        0344  1
;  133        0345  1 REQUIRE 'RTLIN:BPAMSGDEF';
;  134        0407  1
;  135        0408  1 REQUIRE 'RTLIN:BPAFUNDEF';
;  136        0658  1
;  137        0659  1 REQUIRE 'RTLIN:BPAERRDEF';
;  138        0855  1
;  139        0856  1 LIBRARY 'RTLSTARLE';
;  140        0857  1
;  141        0858  1 REQUIRE 'RTLIN:RTLPSECT';
;  142        0953  1
;  143        0954  1 REQUIRE 'RTLIN:OTSLNK';
;  144        1383  1
;  145        1384  1 REQUIRE 'RTLML:OTSLUB';
;  146        1524  1
;  147        1525  1 !
;  148        1526  1 ! MACROS:
;  149        1527  1 !
;  150        1528  1 !       NONE
;  151        1529  1 !
;  152        1530  1 ! EQUATED SYMBOLS:
;  153        1531  1 !
;  154        1532  1 !       NONE
;  155        1533  1 !
;  156        1534  1 ! PSECTS:
```

```
;  157          1535  1 !
;  158          1536  1 declare_psects (bpa);
;  159          1537  1 !
;  160          1538  1 ! OWN STORAGE:
;  161          1539  1 !
;  162          1540  1
;  163          1541  1 GLOBAL
;  164          1542  1     bpa$a_msg : $msg_def;                          ! Block with data needed to be
;  165          1543  1
;  166          1544  1                                                    ! saved between calls
;  167          1545  1
;  168          1546  1 OWN
;  169          1547  1     exit_lock : INITIAL (0),               ! One-time flag for exit handler
;  170          1548  1     exit_reason,
;  171          1549  1     exit_block : VECTOR [4] INITIAL (0, 0, 0, 0),
;  172          1550  1     param_string : VECTOR [20, BYTE];              ! Last parameter string read
;  173          1551  1
;  174          1552  1 !
;  175          1553  1 ! EXTERNAL REFERENCES:
;  176          1554  1 !
;  177          1555  1
;  178          1556  1 EXTERNAL ROUTINE
;  179          1557  1     LIB$GET_EF,                                    ! allocate an event flag
;  180          1558  1     LIB$FREE_EF,                                   ! deallocate an event flag
;  181          1559  1     BAS$$CB_POP: JSB_CB_POP NOVALUE,               ! free an ISB/LUB/RAB
;  182          1560  1     bpa$wakeup;                                    ! Routine to wake the process
;  183          1561  1                                                    ! when a message comes.
```

```
1 GLOBAL ROUTINE bpa$mesag (                              ! Main Routine
1         firqb,                                          ! Address of the FIRQB
1         buflen,                                         ! Length of the user's buffer
1         bufadr,                                         ! Address of the user's buffer
1         bytxfr                                  ! Number of bytes actually transfered
1     ) =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine calls one of bpa$mess_dcl, bpa$mess_lcl, bpa$mess_rcv
1 !       and bpa$mess_rem depending on the subfunction code in firqb. If the
1 !       subfunction code is faulty an error is signalled.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       firqb                   =  firqb address
1 !       buflen.rl.v     Length of output data buffer in bytes, 0 to 512.
1 !       bufadr.ra.v     Starting address of data buffer
1 !       bytxfr.ww.r     Number of data bytes received
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       Depending on the subfunction code, various fields in firqb and
1 !       bpa$a_msg are used as input.
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       Depending on the subfunction code, various fields in firqb and
1 !       bpa$a_msg are used as output.
1 !
1 ! ROUTINE VALUE:
1 !
1 !       1  -  success
1 !       ?  -  failure
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2     BEGIN
2
2     MAP
2         firqb : REF $fqb_def;                     ! Defines firqb
2
2 ! Select the right subfunction and perform it.
2
2     CASE .firqb [fqb$b_subfun] FROM fun$k_minsr TO fun$k_maxsr OF
2         SET
2
2         [fun$k_srlcl] :
2             bpa$mess_lcl (.firqb, .buflen, .bufadr);
2
2         [fun$k_srrem] :
2             bpa$mess_rem (.firqb);
```

```
;  242           1619  2
;  243           1620  2         [fun$k_srdcl] :
;  244           1621  2             bpa$mess_dcl (.firqb);
;  245           1622  2
;  246           1623  2         [fun$k_srrcv] :
;  247           1624  2             bpa$mess_rcv (.firqb, .buflen, .bufadr, .bytxfr);
;  248           1625  2
;  249           1626  2         [INRANGE, OUTRANGE] :
;  250           1627  2             RETURN SIGNAL (errerr);
;  251           1628  2         TES;
;  252           1629  2
;  253           1630  2     RETURN 1;
;  254           1631  1     END;                                      ! End of bpa$mesag


                                                                          .TITLE  BPA$MESAG
                                                                          .IDENT  \1-272\

                                                                          .PSECT  _BPA$DATA,NOEXE,  PIC,2

                                                        00# 00000 BPA$A_MSG::
                                                                          .BYTE   0[562]                                      ;
                                                            00232         .BLKB   2
                                                  00000000  00234 EXIT_LOCK:
                                                                          .LONG   0                                           ;
                                                            00238 EXIT_REASON:
                                                                          .BLKB   4
                    00000000  00000000  00000000  00000000  0023C EXIT_BLOCK:
                                                                          .LONG   0, 0, 0, 0                                  ;
                                                            0024C PARAM_STRING:
                                                                          .BLKB   20

                                                                          .EXTRN  LIB$GET_EF, LIB$FREE_EF
                                                                          .EXTRN  BAS$$CB_POP, BPA$WAKEUP

                                                                          .PSECT  _BPA$CODE,NOWRT,  SHR,  PIC,2

                                                       0004 00000         .ENTRY  BPA$MESAG, Save R2                          ; 1562
                                       52       04   AC  D0 00002         MOVL    FIRQB, R2                                   ; 1611
                         0B       F7   8F       04   A2  8F 00006         CASEB   4(R2), #-9, #11                             ;
         0018          0018          0018         0018      0000C 1$:     .WORD   2$-1$,-                                     ;
         0018          0018          0018         0018      00014                 2$-1$,-                                     ;
         0045          003C          0033         0026      0001C                 2$-1$,-                                     ;
                                                                                  2$-1$,-                                     ;
                                                                                  2$-1$,-                                     ;
                                                                                  2$-1$,-                                     ;
                                                                                  2$-1$,-                                     ;
                                                                                  2$-1$,-                                     ;
                                                                                  3$-1$,-                                     ;
                                                                                  4$-1$,-                                     ;
                                                                                  5$-1$,-                                     ;
                                                                                  6$-1$                                       ;
                                          001A8210   8F  DD 00024 2$:     PUSHL   #1737232                                    ; 1627
                            00000000G  00            01  FB 0002A         CALLS   #1, LIB$SIGNAL                              ;
                                                         04 00031         RET                                                 ;
                                       7E       08   AC  7D 00032 3$:     MOVQ    BUFLEN, -(SP)                               ; 1615
                                                     52  DD 00036         PUSHL   R2                                          ;
                    0000V  CF             03  FB 00038        CALLS   #3, BPA$MESS_LCL                   ;
                                          20  11 0003D        BRB     7$                                 ;
                                          52  DD 0003F  4$:   PUSHL   R2                                 ; 1618
                    0000V  CF             01  FB 00041        CALLS   #1, BPA$MESS_REM                   ;
                                          17  11 00046        BRB     7$                                 ;
                                          52  DD 00048  5$:   PUSHL   R2                                 ; 1621
                    0000V  CF             01  FB 0004A        CALLS   #1, BPA$MESS_DCL                   ;
                                          0E  11 0004F        BRB     7$                                 ;
                           7E        0C   AC  7D 00051  6$:   MOVQ    BUFADR, -(SP)                      ; 1624
                                     08   AC  DD 00055        PUSHL   BUFLEN                             ;
                                          52  DD 00058        PUSHL   R2                                 ;
                    0000V  CF             04  FB 0005A        CALLS   #4, BPA$MESS_RCV                   ;
                           50             01  D0 0005F  7$:   MOVL    #1, R0                             ; 1630
                                              04 00062        RET                                        ; 1631

; Routine Size:  99 bytes,    Routine Base:  _BPA$CODE + 0000

;  255          1632  1
```

```
1 ROUTINE bpa$mess_dcl (firqb) =                          ! Simulates declare receiver
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       A check is made that the caller isn't already a receiver and then
1 !       the input in firqb is checked. If the receivername given in firqb
1 !       isn't already in use, a mailbox is created with logical name equal
1 !       to the receivername prefixed by 'BPA$'and a buffer pointer is
1 !       zeroed to indicate that no partial message is outstanding.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       firqb                                 =  firqb address
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       firqb [fqb$t_rcvnam]                  =  receiver logical name in ASCII,
1 !                                                padded with zero (6 bytes)
1 !       firqb [fqb$b_access]                  =  1 (access type)
1 !       firqb [fqb$w_bmax]                    =  max buffer area to be used for
1 !                                                the receive queue.
1 !                                                >0 => system memory is to be used
1 !                                                <0 => user I/O buffer space is to
1 !                                                      be used
1 !       firqb [20, B_] - firqb [31, B_]       =  0
1 !       bpa$a_msg [msg$l_messchan]            =  0  (indicates that the caller
1 !                                                   isn't a receiver already)
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       bpa$a_msg [msg$l_messchan]            =  channel opened to the mailbox
1 !       bpa$a_msg [msg$t_rcvname]             =  logical name of the mailbox
1 !       bpa$a_msg [msg$t_namedesc]            =  char. string desc. pointing to
1 !                                                logical name of mailbox
1 !       bpa$a_msg [msg$l_buffptr]             =  0 (pointer to the receive buffer)
1 !
1 ! ROUTINE VALUE:
1 !
1 !       1  -  success
1 !       ?  -  failure
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2   BEGIN
2
2   MAP
2       firqb : REF $fqb_def;                           ! Defines firqb
2
2   LOCAL
2       sts,                                            ! Status received from calls
2                                                       ! to system services
2       rslbuf : VECTOR [2, LONG],                      ! Char. string descriptor
```

```
2                                                  ! pointing to $trnlog - buffer
2         rslnam : BLOCK [63, BYTE],               ! Buffer for $trnlog
2         pribuf : VECTOR [2, LONG],               ! Char. string descriptor
2                                                  ! pointing to $getchn - buffer
2         devbuf : BLOCK [200, BYTE],              ! Buffer for $getchn
2         devnam_desc : VECTOR [2, LONG],          ! Char. string descr. pointing
2                                                  ! to the mailbox device name
2         devnam : BLOCK [9, BYTE],                ! Mailbox device name
2         fao_desc : VECTOR [2, LONG];             ! Char. string desc. for
2
2                                                  ! commands to $fao
2
2     LITERAL
2         k_maxmsg_size = 512 + 20,                ! max. message size.
2         k_lrg_sndrcv = 22;                       ! Large send/receive.
2
2     BIND
2         chan = bpa$a_msg [msg$l_messchan],       ! Channel
2         lognam_desc = bpa$a_msg [msg$t_namedesc],        ! Char.descr. of
2                                                  ! receiver name.
2         lognam = bpa$a_msg [msg$t_rcvname],      ! Receiver name
2                                                  ! $qiow
2         buff_ptr = bpa$a_msg [msg$l_bufptr];     ! Buffer pointer
2 ! Is the caller already a receiver? If so, signal an error.
2 ! If this is small send and receive then just return.
2
2     IF .chan NEQU 0
2     THEN
2
2         IF .firqb [fqb$b_function] EQL k_lrg_sndrcv
2         THEN
2             RETURN SIGNAL (inuse)
2         ELSE
2             RETURN 1;
2
2 ! Check input data.
2
2     IF .firqb [fqb$b_access] NEQU 1 OR NOT CH$FAIL (CH$FIND_NOT_CH (12,
2                 firqb [20, b_], 0))
2     THEN
2         RETURN SIGNAL (badfuo);
2
2 ! Create a logical name for the mailbox (receiver name from firqb, prefixed
2 ! by 'BPA$').
2     mess_rcvname (lognam_desc, lognam, .firqb);
2 ! Check that the logical name isn't already in use.
2     rslbuf [0] = 63;
2     rslbuf [1] = rslnam;
2
2     IF (sts = $trnlog (lognam = lognam_desc, rslbuf = rslbuf)) EQLU ss$_normal
2     THEN
2         RETURN SIGNAL (fiexst);
2
2 ! Create a mailbox. If buffer maximum is negative, the mailbox should be
2 ! temporary, otherwise permanent.
2
```

```
;  371       1747 2        IF .firqb [fqb$w_bmax] LEQ 0
;  372       1748 2        THEN
;  373     P 1749 2            sts = $crembx (prmflg = 0, chan = chan,
;  374       1750 3                maxmsg = k_maxmsg_size, bufquo = 0)
;  375       1751 2        ELSE
;  376     P 1752 2            sts = $crembx (prmflg = 1, chan = chan, maxmsg = k_maxmsg_size,
;  377       1753 2                bufquo = 0);
;  378       1754 2
;  379       1755 2        IF NOT .sts THEN RETURN SIGNAL (.sts);
;  380       1756 2
;  381       1757 2      ! Get the device name of the mailbox.
;  382       1758 2        pribuf [0] = 200;
;  383       1759 2        pribuf [1] = devbuf;
;  384       1760 2
;  385       1761 3        IF NOT (sts = $getchn (chan = .chan, pribuf = pribuf))
;  386       1762 2        THEN
;  387       1763 2            RETURN SIGNAL (.sts);
;  388       1764 2
;  389       1765 2        fao_desc [1] = UPLIT BYTE(%ASCIC'!AC!ZW:') + 1;
;  390       1766 2        fao_desc [0] = CH$RCHAR (.fao_desc [1] - 1);
;  391       1767 2        devnam_desc [1] = devnam;
;  392       1768 2        devnam_desc [0] = 9;
;  393       1769 2
;  394     P 1770 3        IF NOT (sts = $fao (fao_desc, devnam_desc [0], devnam_desc,
;  395       1771 3                devbuf [.devbuf [dib$w_devnamoff], a_], .devbuf [dib$w_unit]))
;  396       1772 2        THEN
;  397       1773 2            RETURN SIGNAL (.sts);
;  398       1774 2
;  399       1775 2      ! Assign the logical name to the devicename of the mailbox
;  400       1776 2
;  401       1777 3        IF NOT (sts = $crelog (eqlnam = devnam_desc, lognam = lognam_desc))
;  402       1778 2        THEN
;  403       1779 3            BEGIN
;  404       1780 3      !+
;  405       1781 3      ! If we cannot create a logical then we must delete the permanent mailbox
;  406       1782 3      ! that we created above.
;  407       1783 3      !-
;  408       1784 4            BEGIN
;  409       1785 4
;  410       1786 5            IF .firqb [fqb$w_bmax] LSS 0 THEN $delmbx (chan = chan)
;  411       1787 5
;  412       1788 3            END;
;  413       1789 3            RETURN SIGNAL (prviol, 0, .sts);
;  414       1790 2            END;
;  415       1791 2
;  416       1792 2      ! Prepare for the receive calls by clearing the buffer pointer used by
;  417       1793 2      ! bpa$mess_rcv.
;  418       1794 2        buff_ptr = 0;
;  419       1795 2      !+
;  420       1796 2      ! Declare an exit handler, to delete the name when the image exits.
;  421       1797 2      ! This may be needed if the user's program fails and it does not
;  422       1798 2      ! delete the name itself.
;  423       1799 2      !-
;  424       1800 3        BEGIN
;  425       1801 3
;  426       1802 3        LOCAL
;  427       1803 3            ast_status,
;  428          1804  3         dclexh_status;
;  429          1805  3
;  430          1806  3     ast_status = $setast (enbflg = 0);
;  431          1807  3
;  432          1808  4     IF ( NOT .exit_lock)
;  433          1809  3     THEN
;  434          1810  4         BEGIN
;  435          1811  4         exit_block [1] = exit_handler;
;  436          1812  4         exit_block [2] = 1;
;  437          1813  4         exit_block [3] = exit_reason;
;  438          1814  4         dclexh_status = $dclexh (desblk = exit_block);
;  439          1815  4         exit_lock = 1;
;  440          1816  4         END
;  441          1817  3     ELSE
;  442          1818  3         dclexh_status = 1;
;  443          1819  3
;  444          1820  3     IF (.ast_status EQL ss$_wasset) THEN $setast (enbflg = 1);
;  445          1821  3
;  446          1822  3     IF ( NOT .dclexh_status) THEN SIGNAL (badfuo, 0, .dclexh_status);
;  447          1823  3
;  448          1824  2     END;
;  449          1825  2     RETURN 1;
;  450          1826  1     END;                                            !End of bpa$mess_dcl


                            3A  57  5A  21  43  41  21  07  00063 P.AAA:  .ASCII  <7>\!AC!ZW:\                                 ;

                                                                  CHAN=               BPA$A_MSG
                                                                  LOGNAM_DESC=        BPA$A_MSG+4
                                                                  LOGNAM=             BPA$A_MSG+12
                                                                  BUFF_PTR=           BPA$A_MSG+554
                                                                          .EXTRN  SYS$TRNLOG, SYS$CREMBX
                                                                          .EXTRN  SYS$GETCHN, SYS$FAO
                                                                          .EXTRN  SYS$CRELOG, SYS$DELMBX
                                                                          .EXTRN  SYS$SETAST, SYS$DCLEXH

                                                       007C 00000 BPA$MESS_DCL:
                                                                          .WORD   Save R2,R3,R4,R5,R6                          ; 1633
                                       56 00000000G  00  9E 00002         MOVAB   SYS$SETAST, R6                               ;
                                       55 00000000G  00  9E 00009         MOVAB   LIB$SIGNAL, R5                               ;
                                       54 00000000'  EF  9E 00010         MOVAB   CHAN, R4                                     ;
                                       5E      FECC  CE  9E 00017         MOVAB   -308(SP), SP                                 ;
                                                     64  D5 0001C         TSTL    CHAN                                         ; 1717
                                                     15  13 0001E         BEQL    2$                                           ;
                                       50        04  AC  D0 00020         MOVL    FIRQB, R0                                    ; 1720
                                       16        03  A0  91 00024         CMPB    3(R0), #22                                   ;
                                                     03  13 00028         BEQL    1$                                           ;
                                                   0177  31 0002A         BRW     18$                                          ;
                                          001A8018   8F  DD 0002D 1$:     PUSHL   #1736728                                     ; 1722
                                                     56  11 00033         BRB     6$                                           ;
                                       52        04  AC  D0 00035 2$:     MOVL    FIRQB, R2                                    ; 1728
                                       01        0D  A2  91 00039         CMPB    13(R2), #1                                   ;
                                                     0D  12 0003D         BNEQ    4$                                           ;
                    14   A2            0C            00  3B 0003F         SKPC    #0, #12, 20(R2)                              ; 1729
                                                     02  12 00044         BNEQ    3$                                           ;
                                                     51  D4 00046         CLRL    R1                                           ;
```

```
                          51  D5 00048 3$:     TSTL    R1                            ;
                          08  13 0004A         BEQL    5$                            ;
               001A8090   8F  DD 0004C 4$:     PUSHL   #1736848                      ; 1731
                          37  11 00052         BRB     6$                            ;
                          52  DD 00054 5$:     PUSHL   R2                            ; 1735
                     0C   A4  9F 00056         PUSHAB  LOGNAM                        ;
                     04   A4  9F 00059         PUSHAB  LOGNAM_DESC                   ;
    0000V  CF             03  FB 0005C         CALLS   #3, MESS_RCVNAME              ;
       F8  AD             3F  D0 00061         MOVL    #63, RSLBUF                   ; 1737
       FC  AD        B8   AD  9E 00065         MOVAB   RSLNAM, RSLBUF+4              ; 1738
                          7E  7C 0006A         CLRQ    -(SP)                         ; 1740
                          7E  D4 0006C         CLRL    -(SP)                         ;
                     F8   AD  9F 0006E         PUSHAB  RSLBUF                        ;
                          7E  D4 00071         CLRL    -(SP)                         ;
                     04   A4  9F 00073         PUSHAB  LOGNAM_DESC                   ;
00000000G  00             06  FB 00076         CALLS   #6, SYS$TRNLOG                ;
           53             50  D0 0007D         MOVL    R0, STS                       ;
           01             53  D1 00080         CMPL    STS, #1                       ;
                          09  12 00083         BNEQ    7$                            ;
               001A8080   8F  DD 00085         PUSHL   #1736832                      ; 1742
                        0089  31 0008B 6$:     BRW     11$                           ;
                     0E   A2  B5 0008E 7$:     TSTW    14(R2)                        ; 1747
                          0F  14 00091         BGTR    8$                            ;
                          7E  7C 00093         CLRQ    -(SP)                         ; 1750
                          7E  7C 00095         CLRQ    -(SP)                         ;
           7E      0214   8F  3C 00097         MOVZWL  #532, -(SP)                   ;
                          54  DD 0009C         PUSHL   R4                            ;
                          7E  D4 0009E         CLRL    -(SP)                         ;
                          0D  11 000A0         BRB     9$                            ;
                          7E  7C 000A2 8$:     CLRQ    -(SP)                         ; 1753
                          7E  7C 000A4         CLRQ    -(SP)                         ;
           7E      0214   8F  3C 000A6         MOVZWL  #532, -(SP)                   ;
                          54  DD 000AB         PUSHL   R4                            ;
                          01  DD 000AD         PUSHL   #1                            ;
00000000G  00             07  FB 000AF 9$:     CALLS   #7, SYS$CREMBX                ;
           53             50  D0 000B6         MOVL    R0, STS                       ;
           59             53  E9 000B9         BLBC    STS, 10$                      ; 1755
       B0  AD        C8   8F  9A 000BC         MOVZBL  #200, PRIBUF                  ; 1758
       B4  AD        1C   AE  9E 000C1         MOVAB   DEVBUF, PRIBUF+4              ; 1759
                          7E  7C 000C6         CLRQ    -(SP)                         ; 1761
                     B0   AD  9F 000C8         PUSHAB  PRIBUF                        ;
                          7E  D4 000CB         CLRL    -(SP)                         ;
                          64  DD 000CD         PUSHL   CHAN                          ;
00000000G  00             05  FB 000CF         CALLS   #5, SYS$GETCHN                ;
           53             50  D0 000D6         MOVL    R0, STS                       ;
           39             53  E9 000D9         BLBC    STS, 10$                      ;
       04  AE      FF19   CF  9E 000DC         MOVAB   P.AAA+1, FAO_DESC+4           ; 1765
           50        04   AE  D0 000E2         MOVL    FAO_DESC+4, R0                ; 1766
           6E        FF   A0  9A 000E6         MOVZBL  -1(R0), FAO_DESC              ;
       18  AE        08   AE  9E 000EA         MOVAB   DEVNAM, DEVNAM_DESC+4         ; 1767
       14  AE             09  D0 000EF         MOVL    #9, DEVNAM_DESC               ; 1768
           7E        28   AE  3C 000F3         MOVZWL  DEVBUF+12, -(SP)              ; 1771
           50        2E   AE  3C 000F7         MOVZWL  DEVBUF+14, R0                 ;
                     20 AE40  9F 000FB         PUSHAB  DEVBUF[R0]                    ;
                     1C   AE  9F 000FF         PUSHAB  DEVNAM_DESC                   ;
                     20   AE  9F 00102         PUSHAB  DEVNAM_DESC                   ;
                     10   AE  9F 00105         PUSHAB  FAO_DESC                      ;
```

```
00000000G  00            05  FB 00108        CALLS   #5, SYS$FAO                         ;
           53            50  D0 0010F        MOVL    R0, STS                             ;
           06            53  E8 00112        BLBS    STS, 12$                            ;
                         53  DD 00115 10$:   PUSHL   STS                                 ; 1773
           65            01  FB 00117 11$:   CALLS   #1, LIB$SIGNAL                      ;
                             04 0011A        RET                                         ;
                         7E  D4 0011B 12$:   CLRL    -(SP)                               ; 1777
                     18  AE  9F 0011D        PUSHAB  DEVNAM_DESC                         ;
                     04  A4  9F 00120        PUSHAB  LOGNAM_DESC                         ;
                         7E  D4 00123        CLRL    -(SP)                               ;
00000000G  00            04  FB 00125        CALLS   #4, SYS$CRELOG                      ;
           53            50  D0 0012C        MOVL    R0, STS                             ;
           1C            53  E8 0012F        BLBS    STS, 14$                            ;
                     0E  A2  B5 00132        TSTW    14(R2)                              ; 1786
                         09  18 00135        BGEQ    13$                                 ;
                         54  DD 00137        PUSHL   R4                                  ;
00000000G  00            01  FB 00139        CALLS   #1, SYS$DELMBX                      ;
                         53  DD 00140 13$:   PUSHL   STS                                 ; 1789
                         7E  D4 00142        CLRL    -(SP)                               ;
               001A8050  8F  DD 00144        PUSHL   #1736784                            ;
           65            03  FB 0014A        CALLS   #3, LIB$SIGNAL                      ;
                             04 0014D        RET                                         ;
                   022A  C4  D4 0014E 14$:   CLRL    BUFF_PTR                            ; 1794
                         7E  D4 00152        CLRL    -(SP)                               ; 1806
           66            01  FB 00154        CALLS   #1, SYS$SETAST                      ;
           53            50  D0 00157        MOVL    R0, AST_STATUS                      ;
           28      0234  C4  E8 0015A        BLBS    EXIT_LOCK, 15$                      ; 1808
     0240  C4     0000V  CF  9E 0015F        MOVAB   EXIT_HANDLER, EXIT_BLOCK+4          ; 1811
     0244  C4            01  D0 00166        MOVL    #1, EXIT_BLOCK+8                    ; 1812
     0248  C4      0238  C4  9E 0016B        MOVAB   EXIT_REASON, EXIT_BLOCK+12          ; 1813
                   023C  C4  9F 00172        PUSHAB  EXIT_BLOCK                          ; 1814
00000000G  00            01  FB 00176        CALLS   #1, SYS$DCLEXH                      ;
           52            50  D0 0017D        MOVL    R0, DCLEXH_STATUS                   ;
     0234  C4            01  D0 00180        MOVL    #1, EXIT_LOCK                       ; 1815
                         03  11 00185        BRB     16$                                 ; 1808
           52            01  D0 00187 15$:   MOVL    #1, DCLEXH_STATUS                   ; 1818
           09            53  D1 0018A 16$:   CMPL    AST_STATUS, #9                      ; 1820
                         05  12 0018D        BNEQ    17$                                 ;
                         01  DD 0018F        PUSHL   #1                                  ;
           66            01  FB 00191        CALLS   #1, SYS$SETAST                      ;
           0D            52  E8 00194 17$:   BLBS    DCLEXH_STATUS, 18$                  ; 1822
                         52  DD 00197        PUSHL   DCLEXH_STATUS                       ;
                         7E  D4 00199        CLRL    -(SP)                               ;
               001A8090  8F  DD 0019B        PUSHL   #1736848                            ;
           65            03  FB 001A1        CALLS   #3, LIB$SIGNAL                      ;
           50            01  D0 001A4 18$:   MOVL    #1, R0                              ; 1825
                             04 001A7        RET                                         ; 1826

; Routine Size:  424 bytes,    Routine Base:  _BPA$CODE + 006B
```

```
1 ROUTINE mess_rcvname (name_desc, name, firqb) : NOVALUE =          ! Creates a
1                                                      ! logical
1                                                      ! mailbox name
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       A string of receiver name from firqb, prefixed by 'BPA$' and  padded
1 !       with zeroes to 10 bytes, is given to the second parameter.
1 !       The character string descriptor of the string is given to the first
1 !       parameter.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       name_desc               = address to receive the char. string desc. of
1 !                                 mailbox name
1 !       name                    = address to receive the mailbox name string
1 !                                 (max. 10 bytes)
1 !       firqb                   = address to firqb
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       firqb [fqb$t_rcvnam]  =  receiver logical name in ASCII,
1 !                                padded with zero (6 bytes)
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       name_desc               = char. string desc. of mailbox name
1 !       name                    = string of 'BPA$' + receiver name in firqb,
1 !                                 padded with zero
1 !
1 ! ROUTINE VALUE:
1 !
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2     BEGIN
2
2     MAP
2         name_desc : REF VECTOR [2, LONG],          ! Char. string descr.
2                                                     ! of mailbox name
2         name : REF BLOCK [, BYTE],                  ! Mailbox name
2         firqb : REF $fqb_def;                       ! Defines firqb
2
2     BIND
2         prefix = UPLIT BYTE(%ASCIC'BPA$');         ! First part of mailbox name
2
2 ! Copy 'BPA$' + receiver name from firqb to recname
2     CH$COPY (CH$RCHAR (prefix), prefix + 1, 6, firqb [fqb$t_rcvnam], 0, 10,
2         .name);
2 ! Initialise the character string descriptor
2     name_desc [0] = CH$FIND_CH (10, .name, 0) - .name;
```

```
;  509           1884  2
;  510           1885  2      IF .name_desc [0] LSS 0 THEN name_desc [0] = 10;
;  511           1886  2
;  512           1887  2      name_desc [1] = .name;
;  513           1888  2      RETURN;
;  514           1889  1      END;                                       ! End of mess_rcvname



                                     24  41  50  42  04  00213 P.AAB:   .ASCII  <4>\BPA$\                          ;

                                                               PREFIX=           P.AAB


                                                   03FC  00000 MESS_RCVNAME:
                                                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9       ; 1827
                                     59       F6   AF   9A  00002       MOVZBL  PREFIX, R9                         ; 1880
                                     56       0C   AC   D0  00006       MOVL    FIRQB, R6                          ;
                                     58            0A   D0  0000A       MOVL    #10, R8                            ;
                                     57       08   AC   D0  0000D       MOVL    NAME, R7                           ; 1881
          58          00        E7   AF            59   2C  00011       MOVC5   R9, PREFIX+1, #0, R8, (R7)         ;
                                                   67       00017                                                  ;
                                                   0D   18  00018       BGEQ    1$                                 ;
                                     57            59   C0  0001A       ADDL2   R9, R7                             ;
                                     58            59   C2  0001D       SUBL2   R9, R8                             ;
          58          00        06   A6            06   2C  00020       MOVC5   #6, 6(R6), #0, R8, (R7)            ;
                                                   67       00026                                                  ;
                                     52       04   AC   D0  00027 1$:   MOVL    NAME_DESC, R2                      ; 1883
                 08   BC             0A            00   3A  0002B       LOCC    #0, #10, @NAME                     ;
                                                   02   12  00030       BNEQ    2$                                 ;
                                                   51   D4  00032       CLRL    R1                                 ;
                      62             51       08   AC   C3  00034 2$:   SUBL3   NAME, R1, (R2)                     ;
                                                   03   18  00039       BGEQ    3$                                 ; 1885
                                     62            0A   D0  0003B       MOVL    #10, (R2)                          ;
                                04   A2       08   AC   D0  0003E 3$:   MOVL    NAME, 4(R2)                        ; 1887
                                                        04  00043       RET                                        ; 1889

; Routine Size:  68 bytes,    Routine Base:  _BPA$CODE + 0218
```

```
1 ROUTINE bpa$mess_lcl (                        ! Simulate send local data message
1           firqb,                                           ! Address of FIRQB
1           buflen,                                          ! Length of buffer
1           bufadr                                           ! Address of buffer
1       ) =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !         The receiver mailbox name is created as receiver name from firqb,
1 !         prefixed by 'BPA$', and a channel is assigned to the mailbox.
1 !         The input in firqb is checked and if faulty an error is signalled. The
1 !         parameterstring and the databuffer are copied into a buffer and sent
1 !         to the mailbox. Finally the channel is deassigned.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !         firqb                                   =  address to firqb
1 !         buflen.rl.v            Length of output data buffer in bytes
1 !         bufadr.ra.v            Starting address of data buffer
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !         firqb [fqb$b_jobnr]                     =  0
1 !         firqb [fqb$t_rcvnam]                    =  receiver logical name in ASCII,
1 !                                                    padded with zero (6 bytes)
1 !         firqb [fqb$t_par_str]                   =  optional user parameter string,
1 !                                                    padded with zero (20 bytes)
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !         NONE
1 !
1 ! ROUTINE VALUE:
1 !
1 !         1  -  success
1 !         ?  -  failure
1 !
1 ! SIDE EFFECTS:
1 !
1 !         NONE
1 !
1 !--
1
2     BEGIN
2
2     MAP
2         firqb : REF $fqb_def;                           ! Defines firqb
2
2     LOCAL
2         msg_buffer : BLOCK [532, BYTE],                 ! Buffer for output message
2         sts,                                            ! Status received from calls
2                                                         ! to system services
2         lognam_desc : VECTOR [2, LONG],                 ! Char. string descr. pointing
2                                                         ! to the logical mailbox name
2         lognam : BLOCK [10, BYTE],                      ! Logical name of mailbox
2         chan;                                           ! Channel to mailbox
```

```
;  573        1947  2
;  574        1948  2 ! Create the logical name of the mailbox
;  575        1949  2    mess_rcvname (lognam_desc, lognam, .firqb);
;  576        1950  2 ! Assign a channnel to the mailbox
;  577        1951  2
;  578        1952  3     IF NOT (sts = $assign (devnam = lognam_desc, chan = chan))
;  579        1953  2     THEN
;  580        1954  3         BEGIN
;  581        1955  3
;  582        1956  3         IF .sts EQLU %X'144'
;  583        1957  3         THEN
;  584        1958  3             RETURN SIGNAL (nosuch)
;  585        1959  3         ELSE
;  586        1960  3             RETURN SIGNAL (.sts);
;  587        1961  3
;  588        1962  2         END;
;  589        1963  2
;  590        1964  2 ! Check input data.
;  591        1965  2
;  592        1966  2     IF (.buflen GTRU 512) THEN RETURN SIGNAL (badfuo);
;  593        1967  2
;  594        1968  2     IF .firqb [fqb$b_jobnr] NEQU 0 THEN RETURN SIGNAL (errerr);
;  595        1969  2
;  596        1970  2 ! Fill msg_buffer from parameter string and data buffer
;  597        1971  2    CH$COPY (20, firqb [fqb$t_par_str], .buflen, .bufadr, 0, 532, msg_buffer);
;  598        1972  2 ! Send message to mailbox
;  599        1973  2    bpa$mess_send (.chan, msg_buffer, 20 + .buflen);
;  600        1974  2 ! Deassign the channel
;  601        1975  2
;  602        1976  2     IF NOT (sts = $dassgn (chan = .chan)) THEN RETURN SIGNAL (.sts);
;  603        1977  2
;  604        1978  2     RETURN 1;
;  605        1979  1     END;                                        !End of bpa$mess_lcl



                                                                 .EXTRN  SYS$ASSIGN, SYS$DASSGN

                                            01FC 00000 BPA$MESS_LCL:
                                                                 .WORD   Save R2,R3,R4,R5,R6,R7,R8                 ; 1890
                    5E       FDD4    CE   9E 00002               MOVAB   -556(SP), SP                              ;
                    52         04    AC   D0 00007               MOVL    FIRQB, R2                                 ; 1949
                                     52   DD 0000B               PUSHL   R2                                        ;
                               08    AE   9F 0000D               PUSHAB  LOGNAM                                    ;
                               18    AE   9F 00010               PUSHAB  LOGNAM_DESC                               ;
           A5       AF               03   FB 00013               CALLS   #3, MESS_RCVNAME                          ;
                                     7E   7C 00017               CLRQ    -(SP)                                     ; 1952
                               08    AE   9F 00019               PUSHAB  CHAN                                      ;
                               1C    AE   9F 0001C               PUSHAB  LOGNAM_DESC                               ;
     00000000G      00               04   FB 0001F               CALLS   #4, SYS$ASSIGN                            ;
                    58               50   D0 00026               MOVL    R0, STS                                   ;
                    11               58   E8 00029               BLBS    STS, 1$                                   ;
     00000144       8F               58   D1 0002C               CMPL    STS, #324                                 ; 1956
                                     66   12 00033               BNEQ    5$                                        ;
                         001A8028    8F   DD 00035               PUSHL   #1736744                                  ; 1958
                                     60   11 0003B               BRB     6$                                        ;
     00000200       8F         08    AC   D1 0003D 1$:           CMPL    BUFLEN, #512                              ; 1966
```

```
                                                       08   1B 00045          BLEQU    2$                                       ;
                                            001A8090   8F   DD 00047          PUSHL    #1736848                                 ;
                                                       4E   11 0004D          BRB      6$                                       ;
                                                  05   A2   95 0004F 2$:      TSTB     5(R2)                                    ; 1968
                                                       08   13 00052          BEQL     3$                                       ;
                                            001A8210   8F   DD 00054          PUSHL    #1737232                                 ;
                                                       41   11 0005A          BRB      6$                                       ;
                                        57      0214   8F   3C 0005C 3$:      MOVZWL   #532, R7                                 ; 1971
                                        56        18   AE   9E 00061          MOVAB    MSG_BUFFER, R6                           ;
           57             00       0C   A2             14   2C 00065          MOVC5    #20, 12(R2), #0, R7, (R6)                ;
                                                       66      0006B                                                            ;
                                                       0E   18 0006C          BGEQ     4$                                       ;
                                        56             14   C0 0006E          ADDL2    #20, R6                                  ;
                                        57             14   C2 00071          SUBL2    #20, R7                                  ;
           57             00       0C   BC        08   AC   2C 00074          MOVC5    BUFLEN, @BUFADR, #0, R7, (R6)            ;
                                                       66      0007B                                                            ;
                          7E       08   AC             14   C1 0007C 4$:      ADDL3    #20, BUFLEN, -(SP)                       ; 1973
                                                  1C   AE   9F 00081          PUSHAB   MSG_BUFFER                               ;
                                                  08   AE   DD 00084          PUSHL    CHAN                                     ;
                                   0000V   CF             03   FB 00087          CALLS    #3, BPA$MESS_SEND                        ;
                                                       6E   DD 0008C          PUSHL    CHAN                                     ; 1976
                               00000000G   00             01   FB 0008E          CALLS    #1, SYS$DASSGN                           ;
                                        58             50   D0 00095          MOVL     R0, STS                                  ;
                                        0A             58   E8 00098          BLBS     STS, 7$                                  ;
                                                       58   DD 0009B 5$:      PUSHL    STS                                      ;
                               00000000G   00             01   FB 0009D 6$:      CALLS    #1, LIB$SIGNAL                           ;
                                                            04 000A4          RET                                               ;
                                        50             01   D0 000A5 7$:      MOVL     #1, R0                                   ; 1978
                                                            04 000A8          RET                                               ; 1979

; Routine Size:  169 bytes,    Routine Base:  _BPA$CODE + 025C
;  607          1980   1 ROUTINE bpa$mess_send (chan, p1, p2) : NOVALUE =         ! Sends message
;  608          1981   1                                                   ! to a mailbox
;  609          1982   1
;  610          1983   1 !++
;  611          1984   1 ! FUNCTIONAL DESCRIPTION:
;  612          1985   1 !
;  613          1986   1 !       Sends a message to a mailbox using $qiow and checks the return
;  614          1987   1 !       status.
;  615          1988   1 !
;  616          1989   1 ! FORMAL PARAMETERS:
;  617          1990   1 !
;  618          1991   1 !       chan  =  channel number opened to mailbox
;  619          1992   1 !       p1    =  pointer to databuffer
;  620          1993   1 !       p2    =  number of bytes to send
;  621          1994   1 !
;  622          1995   1 ! IMPLICIT INPUTS:
;  623          1996   1 !
;  624          1997   1 !       NONE
;  625          1998   1 !
;  626          1999   1 ! IMPLICIT OUTPUTS:
;  627          2000   1 !
;  628          2001   1 !       NONE
;  629          2002   1 !
;  630          2003   1 ! ROUTINE VALUE:
;  631          2004   1 !
;  632          2005   1 !       ?  -  failure
;  633          2006   1 !
;  634          2007   1 ! SIDE EFFECTS:
;  635          2008   1 !
;  636          2009   1 !       NONE
;  637          2010   1 !
;  638          2011   1 !--
;  639          2012   1
;  640          2013   2     BEGIN
;  641          2014   2
;  642          2015   2     LOCAL
;  643          2016   2         sts,                                ! Status received from calls to system
;  644          2017   2         status,                             ! need 2 locals for status
;  645          2018   2         event_flag,
;  646          2019   2         ! services
;  647          2020   2         iosb : BLOCK [8, BYTE];             ! Status block from $qiow               ! M 260
;  648          2021   2 !+
;  649          2022   2 ! Allocate an event flag.
;  650          2023   2 !-
;  651          2024   2
;  652          2025   2     sts = LIB$GET_EF (event_flag);
;  653          2026   2     IF (NOT .sts) THEN RETURN SIGNAL (.sts);
;  654          2027   2
;  655          2028   2 ! Send the message
;  656          2029   2
;  657        P 2030   2     sts = $qiow (efn = .event_flag, chan = .chan, iosb = iosb,
;  658          2031   2           func = io$_writevblk OR io$m_now, p1 = .p1, p2 = .p2);
;  659          2032   2 !+
;  660          2033   2 ! Make sure event flag is deallocated.
;  661          2034   2 !-
;  662          2035   2
;  663          2036   2         status = LIB$FREE_EF (event_flag);
; 664       2037 2       IF (NOT .status) THEN RETURN SIGNAL (.status);
; 665       2038 2
; 666       2039 2 !+
; 667       2040 2 ! Now check return status from $QIOW.
; 668       2041 2 !-
; 669       2042 3       IF NOT (.sts)
; 670       2043 2       THEN
; 671       2044 3          BEGIN
; 672       2045 3             IF .sts EQLU %X'19C'
; 673       2046 3             THEN
; 674       2047 3                RETURN SIGNAL (noroom)
; 675       2048 3             ELSE
; 676       2049 3                RETURN SIGNAL (.sts);
; 677       2050 3
; 678       2051 2          END;
; 679       2052 2
; 680       2053 2
; 681       2054 2 ! Check status returned in iosb
; 682       2055 2
; 683       2056 2    IF NOT .iosb [0, w_] THEN RETURN SIGNAL (.iosb [0, w_]);
; 684       2057 2
; 685       2058 2    RETURN;
; 686       2059 1    END;                              ! End of bpa$mess_send


                                                         .EXTRN  SYS$QIOW

                                         0004 00000 BPA$MESS_SEND:
                                                         .WORD   Save R2                        ; 1980
                               5E       0C C2 00002      SUBL2   #12, SP                        ;
                                        5E DD 00005      PUSHL   SP                             ; 2025
                     00000000G 00       01 FB 00007      CALLS   #1, LIB$GET_EF                 ;
                               52       50 D0 0000E      MOVL    R0, STS                        ;
                               04       52 E8 00011      BLBS    STS, 2$                        ; 2026
                                        52 DD 00014 1$:  PUSHL   STS                            ;
                                        4D 11 00016      BRB     5$                             ;
                                        7E 7C 00018 2$:  CLRQ    -(SP)                          ; 2031
                                        7E 7C 0001A      CLRQ    -(SP)                          ;
                               7E   08  AC 7D 0001C      MOVQ    P1, -(SP)                      ;
                                        7E 7C 00020      CLRQ    -(SP)                          ;
                                    24  AE 9F 00022      PUSHAB  IOSB                           ;
                               7E   70  8F 9A 00025      MOVZBL  #112, -(SP)                    ;
                                    04  AC DD 00029      PUSHL   CHAN                           ;
                                    2C  AE DD 0002C      PUSHL   EVENT_FLAG                     ;
                     00000000G 00       0C FB 0002F      CALLS   #12, SYS$QIOW                  ;
                               52       50 D0 00036      MOVL    R0, STS                        ;
                                        5E DD 00039      PUSHL   SP                             ; 2036
                     00000000G 00       01 FB 0003B      CALLS   #1, LIB$FREE_EF                ;
                               04       50 E8 00042      BLBS    STATUS, 3$                     ; 2037
                                        50 DD 00045      PUSHL   STATUS                         ;
                                        1C 11 00047      BRB     5$                             ;
                               11       52 E8 00049 3$:  BLBS    STS, 4$                        ; 2042
                     0000019C  8F       52 D1 0004C      CMPL    STS, #412                      ; 2045
                                        BF 12 00053      BNEQ    1$                             ;
                              001A8020  8F DD 00055      PUSHL   #1736736                       ; 2047
                                        08 11 0005B      BRB     5$                             ;
```

```
                   0B      04   AE  E8 0005D 4$:     BLBS    IOSB, 6$                              ; 2056
                   7E      04   AE  3C 00061         MOVZWL  IOSB, -(SP)                           ;
        00000000G  00           01  FB 00065 5$:     CALLS   #1, LIB$SIGNAL                        ;
                                    04 0006C 6$:     RET                                           ; 2059

; Routine Size:  109 bytes,    Routine Base:  _BPA$CODE + 0305
```

```
ROUTINE bpa$mess_rcv (                                  ! Simulates receive message
        firqb,                                          ! Address of FIRQB
        buflen,                                         ! Length of input data buffer
        bufadr,                                 ! Starting address of data buffer
        bytxfr                                  ! The number of bytes transferred
   ) =

!++
! FUNCTIONAL DESCRIPTION:
!
!       A check is made that the caller is a declared receiver and then
!       the input in firqb is checked. If no partial message is outstanding
!       the mailbox is read into a buffer and if a message is received the
!       parameter string is filled and the buffer pointer is moved 20 bytes
!       forward. The caller's data buffer is filled with data from buffer
!       pointer and onwards until the end of the message, and the buffer
!       pointer is updated. If the whole message is transferred to the caller
!       or the caller wants the message truncated, then the buffer pointer
!       is cleared to indicate that no partial message is outstanding.
!
! FORMAL PARAMETERS:
!
!       firqb                               =  firqb address
!       buflen.rl.v             Size of the receive data buffer, in bytes
!       bufadr.ra.v             Starting address of the data buffer
!       bytxfr.ww.r             Number of bytes transferred
!
! IMPLICIT INPUTS:
!
!       firqb [fqb$b_rmod]                  =  receive modifiers :
!                                              bit 0 set -caller wants to wait
!                                                         for messages
!                                              bit 1 set -caller wants message
!                                                         truncated
!       firqb [8, W_] - firqb [10, W_]      =  0
!       firqb [fqb$w_sleep]                 =  sleep time (in seconds)
!       firqb [20, W_] - firqb [31, W_]     =  0
!       bpa$a_msg [msg$l_remain]            =  the number of bytes that remain if
!                                              a partial message is outstanding
!       bpa$a_msg [msg$t_buffer]            =  points to buffer containing the
!                                              remaining bytes to transfer if a
!                                              partial message is outstanding
!       bpa$a_msg [msg$l_messchan]          =  channel opened to mailbox by
!                                              bpa$mess_dcl
!       bpa$a_msg [ms$l_buffptr]            =  points to next byte to be
!                                              transferred in msg_buffer (0 if
!                                              buffer is empty)
!
! IMPLICIT OUTPUTS:
!
!       firqb [fqb$w_not_tran]              =  number of data bytes not
!                                              transferred
!       firqb [fqb$b_subfun]                =  -1                          ! A 261
!       firqb [fqb$t_par_str]               =  data passed as parameters by the
!                                              sender, padded with zero
!       bpa$a_msg [msg$l_remain]            =  the number of bytes that remain if
!                                              a partial message is outstanding
```

```
1 !        bpa$a_msg [msg$t_buffer]          =  contains the remaining bytes to
1 !                                              transfer if a partial message is
1 !                                              outstanding
1 !        bpa$a_msg [msg$l_buffptr]         =  points to next byte to be
1 !                                              transferred in msg_buffer (0 if
1 !                                              buffer is empty)
1 !        firqb [fqb$b_jobnr]               =  0
1 !
1 ! ROUTINE VALUE:
1 !
1 !        1  -  success
1 !        ?  -  failure
1 !
1 ! SIDE EFFECTS:
1 !
1 !        NONE
1 !
1 !--
1
2    BEGIN
2
2    MAP
2        firqb : REF $fqb_def,                    ! Defines firqb
2        bytxfr : REF VECTOR [1, WORD];           ! Where to store byte count
2
2    BIND
2        bytes_remain = bpa$a_msg [msg$l_remain],        ! Remaining bytes
2        buff_ptr = bpa$a_msg [msg$l_bufptr],     ! Buffer pointer
2        rcv_mod = firqb [fqb$b_rmod] : BITVECTOR [8],   ! Receive modifier
2        msg_buffer = bpa$a_msg [msg$t_buffer];  ! Buffer
2
2 ! Is the caller a declared receiver? If not, then signal an error.
2
2    IF .bpa$a_msg [msg$l_messchan] EQLU 0 THEN RETURN SIGNAL (badfuo);
2
2 ! Check input data
2
3    IF ( NOT CH$FAIL (CH$FIND_NOT_CH (4, firqb [8, b_], 0)) OR  !
3        NOT CH$FAIL (CH$FIND_NOT_CH (12, firqb [20, b_], 0)))
2    THEN
2        RETURN SIGNAL (badfuo);
2
2 ! If no partial message is outstanding (i.e. buffer pointer = 0), then try to
2 ! read the mailbox into msg_buffer. The length of the received message is
2 ! stored in bytes_remain.
2
2    IF .buff_ptr EQLU 0
2    THEN
3        BEGIN
3        bytes_remain = read_mailbox (.firqb);
3 ! If a message is received, then fill the parameter buffer
3
3        IF .bytes_remain LEQU 20
3        THEN
4            BEGIN
4            CH$COPY (.bytes_remain, msg_buffer, 0, 20, param_string [0]);
4            bytes_remain = 0;
```

```
;    802            2174  4                 END
;    803            2175  3             ELSE
;    804            2176  4                 BEGIN
;    805            2177  4                 CH$COPY (20, msg_buffer, 0, 20, param_string [0]);
;    806            2178  4                 buff_ptr = msg_buffer + 20;
;    807            2179  4                 bytes_remain = .bytes_remain - 20;
;    808            2180  3                 END;
;    809            2181  3
;    810            2182  2             END;
;    811            2183  2
;    812            2184  2   !+
;    813            2185  2   ! Place the parameter buffer in the FIRQB.  This will be either from the
;    814            2186  2   ! beginning of a message (above) or from the previous message.
;    815            2187  2   !-
;    816            2188  2         CH$COPY (20, param_string [0], 0, 20, firqb [fqb$t_par_str]);
;    817            2189  2   ! If msg-buffer is empty, then clear 'number of bytes transferred' and
;    818            2190  2   ! 'number of bytes not transferred'.
;    819            2191  2
;    820            2192  2         IF .bytes_remain EQLU 0
;    821            2193  2         THEN
;    822            2194  3             BEGIN
;    823            2195  3             bytxfr [0] = 0;
;    824            2196  3             firqb [fqb$w_not_tran] = 0;
;    825            2197  3             END
;    826            2198  2         ELSE
;    827            2199  2   ! If the number of bytes remaining in msg-buffer is less than bufferlength,
;    828            2200  2   ! then set 'number of bytes transferred' equal to number of bytes remaining
;    829            2201  2   ! and clear 'number of bytes not transferred'.
;    830            2202  3             BEGIN
;    831            2203  3
;    832            2204  3             IF .bytes_remain LEQU .buflen
;    833            2205  3             THEN
;    834            2206  4                 BEGIN
;    835            2207  4                 bytxfr [0] = .bytes_remain;
;    836            2208  4                 firqb [fqb$w_not_tran] = 0;
;    837            2209  4                 END
;    838            2210  3             ELSE
;    839            2211  3   ! If the number of bytes remaining in msg-buffer is larger than bufferlength,
;    840            2212  3   ! then set 'number of bytes transferred' equal to bufferlength and 'number of
;    841            2213  3   ! bytes not transferred' equal to number of bytes remaining minus the buffer-
;    842            2214  3   ! length.
;    843            2215  4                 BEGIN
;    844            2216  4                 bytxfr [0] = .buflen;
;    845            2217  4                 firqb [fqb$w_not_tran] = .bytes_remain - .buflen;
;    846            2218  3                 END;
;    847            2219  3
;    848            2220  3   ! Copy 'number of bytes transferred' from msg-buffer to receive buffer and
;    849            2221  3   ! update the buffer pointer.
;    850            2222  3             CH$COPY (.bytxfr [0], .buff_ptr, 0, .buflen, .bufadr);
;    851            2223  3             buff_ptr = .buff_ptr + .bytxfr [0];
;    852            2224  2             END;
;    853            2225  2
;    854            2226  2   ! Set number of bytes remaining in msg-buffer equal to number of bytes not
;    855            2227  2   ! transferred.
;    856            2228  2         bytes_remain = .firqb [fqb$w_not_tran];
;    857            2229  2   ! Set the subfunction code for data message in firqb.                       ! A 261
;    858            2230  2         firqb [fqb$b_subfun] = fun$k_srlcl;          ! A 261
;  859          2231  2 ! If all the message is transferred or if the caller wants the message
;  860          2232  2 ! truncated, clear the bufferpointer.
;  861          2233  2
;  862          2234  2     IF .bytes_remain EQLU 0 OR .rcv_mode [1] THEN buff_ptr = 0;
;  863          2235  2
;  864          2236  2 !+
;  865          2237  2 ! In case the caller is using the job number field, set it to zero.
;  866          2238  2 !-
;  867          2239  2     firqb [fqb$b_jobnr] = 0;
;  868          2240  2     RETURN 1;
;  869          2241  1     END;                                        !End of bpa$mess_rcv



                                                                   BYTES_REMAIN=       BPA$A_MSG+558
                                                                   BUFF_PTR=           BPA$A_MSG+554
                                                                   MSG_BUFFER=         BPA$A_MSG+22


                                                        00FC 00000 BPA$MESS_RCV:
                                                                           .WORD   Save R2,R3,R4,R5,R6,R7                      : 2060
                                        57 00000000'  EF   9E 00002        MOVAB   BYTES_REMAIN, R7                            :
                                        56       04   AC   D0 00009        MOVL    FIRQB, R6                                   : 2145
                                               FDD2   C7   D5 0000D        TSTL    BPA$A_MSG                                   : 2150
                                                      1A   13 00011        BEQL    3$                                          :
                    08   A6             04            00   3B 00013        SKPC    #0, #4, 8(R6)                               : 2154
                                                      02   12 00018        BNEQ    1$                                          :
                                                      51   D4 0001A        CLRL    R1                                          :
                                                      51   D5 0001C 1$:    TSTL    R1                                          :
                                                      0D   12 0001E        BNEQ    3$                                          :
                    14   A6             0C            00   3B 00020        SKPC    #0, #12, 20(R6)                             : 2155
                                                      02   12 00025        BNEQ    2$                                          :
                                                      51   D4 00027        CLRL    R1                                          :
                                                      51   D5 00029 2$:    TSTL    R1                                          :
                                                      0E   13 0002B        BEQL    4$                                          :
                                           001A8090   8F   DD 0002D 3$:    PUSHL   #1736848                                    : 2157
                            00000000G   00            01   FB 00033        CALLS   #1, LIB$SIGNAL                              :
                                                           04 0003A        RET                                                 :
                                                 FC   A7   D5 0003B 4$:    TSTL    BUFF_PTR                                    : 2163
                                                      2C   12 0003E        BNEQ    6$                                          :
                                                      56   DD 00040        PUSHL   R6                                          : 2166
                                0000V   CF            01   FB 00042        CALLS   #1, READ_MAILBOX                            :
                                        67            50   D0 00047        MOVL    R0, BYTES_REMAIN                            :
                                        14            67   D1 0004A        CMPL    BYTES_REMAIN, #20                           : 2169
                                                      0D   1A 0004D        BGTRU   5$                                          :
           14            00      FDE8   C7            67   2C 0004F        MOVC5   BYTES_REMAIN, MSG_BUFFER, #0, #20, -        : 2172
                                                 1E   A7      00056                PARAM_STRING                                :
                                                      67   D4 00058        CLRL    BYTES_REMAIN                                : 2173
                                                      10   11 0005A        BRB     6$                                          : 2169
                    1E   A7      FDE8   C7            14   28 0005C 5$:    MOVC3   #20, MSG_BUFFER, PARAM_STRING               : 2177
                                   FC   A7     FDFC   C7   9E 00063        MOVAB   MSG_BUFFER+20, BUFF_PTR                     : 2178
                                        67            14   C2 00069        SUBL2   #20, BYTES_REMAIN                           : 2179
                    0C   A6        1E   A7            14   28 0006C 6$:    MOVC3   #20, PARAM_STRING, 12(R6)                   : 2188
                                        50            67   D0 00072        MOVL    BYTES_REMAIN, R0                            : 2192
                                                      08   12 00075        BNEQ    7$                                          :
                                                 10   BC   B4 00077        CLRW    @BYTXFR                                     : 2195
                                                 0A   A6   B4 0007A        CLRW    10(R6)                                      : 2196
```

```
                                                      2C  11 0007D         BRB     10$                                      ; 2192
                                   08   AC            50  D1 0007F 7$:     CMPL    R0, BUFLEN                               ; 2204
                                                      09  1A 00083         BGTRU   8$                                       ;
                                   10   BC            50  B0 00085         MOVW    R0, @BYTXFR                              ; 2207
                                                 0A   A6  B4 00089         CLRW    10(R6)                                   ; 2208
                                                      0B  11 0008C         BRB     9$                                       ; 2204
                                   10   BC       08   AC  B0 0008E 8$:     MOVW    BUFLEN, @BYTXFR                          ; 2216
                     0A   A6            50       08   AC  A3 00093         SUBW3   BUFLEN, R0, 10(R6)                       ; 2217
       08   AC            00       FC   B7       10   BC  2C 00099 9$:     MOVC5   @BYTXFR, @BUFF_PTR, #0, BUFLEN, @BUFADR  ; 2222
                                                 0C   BC     000A1                                                          ;
                                        50       10   BC  3C 000A3         MOVZWL  @BYTXFR, R0                              ; 2223
                                   FC   A7            50  C0 000A7         ADDL2   R0, BUFF_PTR                             ;
                                        67       0A   A6  3C 000AB 10$:    MOVZWL  10(R6), BYTES_REMAIN                     ; 2228
                                   04   A6            01  8E 000AF         MNEGB   #1, 4(R6)                                ; 2230
                                                      67  D5 000B3         TSTL    BYTES_REMAIN                             ; 2234
                                                      05  13 000B5         BEQL    11$                                      ;
                          03       05   A6            01  E1 000B7         BBC     #1, 5(R6), 12$                           ;
                                                 FC   A7  D4 000BC 11$:    CLRL    BUFF_PTR                                 ;
                                                 05   A6  94 000BF 12$:    CLRB    5(R6)                                    ; 2239
                                        50            01  D0 000C2         MOVL    #1, R0                                   ; 2240
                                                          04 000C5         RET                                              ; 2241

; Routine Size:  198 bytes,    Routine Base:  _BPA$CODE + 0372
;  871        2242  1 ROUTINE read_mailbox (firqb) =                          ! Reads the mailbox
;  872        2243  1
;  873        2244  1 !++
;  874        2245  1 ! FUNCTIONAL DESCRIPTION:
;  875        2246  1 !
;  876        2247  1 !       The receiver mailbox is read into a buffer. If a message is received,
;  877        2248  1 !       the length of the message is returned to the caller. If no message is
;  878        2249  1 !       received and the caller doesn't want to wait, an error is signalled.
;  879        2250  1 !       If the caller wants to wait, an AST is set to awaken the process when
;  880        2251  1 !       the message comes and the process is put to sleep. If a sleep time is
;  881        2252  1 !       specified in firqb, the process is awakened when the time has expired.
;  882        2253  1 !       When the process finally wakes up either by the arrival of a message
;  883        2254  1 !       or by an expired sleep time, an error is signalled.
;  884        2255  1 !
;  885        2256  1 ! FORMAL PARAMETERS:
;  886        2257  1 !
;  887        2258  1 !       firqb                           =  address to firqb
;  888        2259  1 !
;  889        2260  1 ! IMPLICIT INPUTS:
;  890        2261  1 !
;  891        2262  1 !       bpa$a_msg [msg$l_messchan]  =  channel opened to mailbox by
;  892        2263  1 !                                      bpa$mess_dcl
;  893        2264  1 !       bpa$a_msg [msg$t_buffer]    =  points to buffer for message from
;  894        2265  1 !                                      mailbox
;  895        2266  1 !       firqb [fqb$b_rmod]          =  receive modifier :
;  896        2267  1 !                                      bit 0 set - caller wants to wait for
;  897        2268  1 !                                                  messages
;  898        2269  1 !       firqb [fqb$w_sleep]         =  sleep time (in seconds)
;  899        2270  1 !
;  900        2271  1 ! IMPLICIT OUTPUTS:
;  901        2272  1 !
;  902        2273  1 !       bpa$a_msg [msg$t_buffer]    =  points to buffer for received message
;  903        2274  1 !
;  904        2275  1 ! ROUTINE VALUE:
;  905        2276  1 !
;  906        2277  1 !       length of received message (in bytes)
;  907        2278  1 !
;  908        2279  1 ! SIDE EFFECTS:
;  909        2280  1 !
;  910        2281  1 !       NONE
;  911        2282  1 !
;  912        2283  1 !--
;  913        2284  1
;  914        2285  2     BEGIN
;  915        2286  2
;  916        2287  2     BUILTIN
;  917        2288  2         emul;
;  918        2289  2
;  919        2290  2     global register
;  920        2291  2         ccb = 11: ref block [,byte];
;  921        2292  2
;  922        2293  2     MAP
;  923        2294  2         firqb : REF $fqb_def;                           ! Defines firqb
;  924        2295  2
;  925        2296  2     LOCAL
;  926        2297  2         sts,                                            ! Status received from calls
;  927        2298  2                                                         ! to system services
```

```
    status,                                     ! need 2 locals for status
    event_flag,
    iosb : BLOCK [8, BYTE],         ! Status block from $qiow       ! M 260
    daytim : VECTOR [2, LONG];                  ! Stores sleeptime in system
                                                ! time format

  EXTERNAL
    OTS$$A_CUR_LUB;

  BIND
    chan = bpa$a_msg [msg$l_messchan],          ! Channel
    msg_buffer = bpa$a_msg [msg$t_buffer],      ! Buffer
    qiow_sts = iosb [0, w_] : WORD,             ! Ret.stat.     ! M 260
                                                ! by $qiow
    transf_bytes = iosb [2, w_] : WORD,         ! # of trans-! M 260
                                                ! ferred bytes
    rcv_mod = firqb [fqb$b_rmod] : BITVECTOR [8];   ! Receiver modifier

  CCB = .OTS$$A_CUR_LUB;

!+
! Allocate event flag.
!-

  sts = LIB$GET_EF (event_flag);
  IF (NOT .sts) THEN RETURN SIGNAL (.sts);

! Try to get a new message from the mailbox

  sts = $qiow (efn = .event_flag, chan = .chan, iosb = iosb,
      func = io$_readvblk OR io$m_now, p1 = msg_buffer, p2 = 532);

!+
! Deallocate the event flag, and then check the status of the $QIOW.
!-

  status = LIB$FREE_EF (event_flag);
  IF NOT (.status) THEN RETURN SIGNAL (.status);

  IF NOT (.sts) THEN RETURN SIGNAL (.sts);

  IF NOT .qiow_sts AND .qiow_sts NEQU ss$_endoffile
  THEN
    RETURN SIGNAL (.qiow_sts);

! If a message is received, then return number of transferred bytes.

  IF .qiow_sts NEQU ss$_endoffile THEN RETURN .transf_bytes;

! If there is no message and the caller doesn't want to wait, then signal
! an error.

  IF NOT .rcv_mod [0]
  THEN
    BEGIN
    !+
    ! free up this ISB/LUB/RAB
```

```
;  985          2356  3              !-
;  986          2357  3              IF .CCB NEQ 0
;  987          2358  3              THEN IF .CCB [LUB$V_IO_ACTIVE] THEN BAS$$CB_POP ();
;  988          2359  3              RETURN SIGNAL (nosuch);
;  989          2360  2              END;
;  990          2361  2
;  991          2362  2  ! If the caller wants to wait for the message and a sleeptime is specified,
;  992          2363  2  ! then schedule a wakeup.
;  993          2364  2
;  994          2365  2      IF .firqb [fqb$w_sleep] NEQU 0
;  995          2366  2      THEN
;  996          2367  3          BEGIN
;  997          2368  3          emul (%REF (-10000000), firqb [fqb$w_sleep], %REF (0), daytim [0]);
;  998          2369  3
;  999          2370  3          IF NOT (sts = $schdwk (daytim = daytim)) THEN RETURN SIGNAL (.sts);
; 1000          2371  3
; 1001          2372  3          END
; 1002          2373  2      ELSE
; 1003          2374  2  ! If the caller wants to wait for the message but no sleeptime is specified,
; 1004          2375  2  ! then set an attention AST (so that the process is awakened when the next
; 1005          2376  2  ! message comes).
; 1006          2377  3          BEGIN
; 1007          2378  3
; 1008          2379  3          sts = LIB$GET_EF (event_flag);
; 1009          2380  3          IF (NOT .sts) THEN RETURN SIGNAL (.sts);
; 1010          2381  3
; 1011        P 2382  4          IF NOT (sts = $qiow (efn = .event_flag, chan = .chan, iosb = iosb,
; 1012          2383  4                  func = io$_setmode OR io$m_wrtattn, p1 = bpa$wakeup))
; 1013          2384  3          THEN
; 1014          2385  3              RETURN SIGNAL (.sts);
; 1015          2386  3
; 1016          2387  3          IF NOT .qiow_sts THEN RETURN SIGNAL (.qiow_sts);
; 1017          2388  3
; 1018          2389  3          sts = LIB$FREE_EF (event_flag);
; 1019          2390  3          IF (NOT .sts) THEN RETURN SIGNAL (.sts);
; 1020          2391  3
; 1021          2392  2          END;
; 1022          2393  2
; 1023          2394  2  ! Hibernate the process
; 1024          2395  2      $hiber;
; 1025          2396  2  ! When the process wakes up, return an error.
; 1026          2397  2      RETURN SIGNAL (nosuch);
; 1027          2398  1      END;                                        !End of read_mailbox


                                                 CHAN=                   BPA$A_MSG
                                                 MSG_BUFFER=             BPA$A_MSG+22
                                                          .EXTRN  OTS$$A_CUR_LUB, SYS$SCHDWK
                                                          .EXTRN  SYS$HIBER

                                      08FC 00000 READ_MAILBOX:
                                                          .WORD   Save R2,R3,R4,R5,R6,R7,R11                   ; 2242
                     57 00000000G  00  9E 00002           MOVAB   LIB$FREE_EF, R7                              ;
                     56 00000000G  00  9E 00009           MOVAB   SYS$QIOW, R6                                 ;
                     55 00000000G  00  9E 00010           MOVAB   LIB$GET_EF, R5                               ;
                     54 00000000'  EF  9E 00017           MOVAB   CHAN, R4                                     ;
```

```
                             5E             14  C2 0001E         SUBL2   #20, SP                              ;
                             52        04   AC  D0 00021         MOVL    FIRQB, R2                            ; 2315
                             5B 00000000G   00  D0 00025         MOVL    OTS$$A_CUR_LUB, CCB                  ; 2317
                                            5E  DD 0002C         PUSHL   SP                                   ; 2323
                             65             01  FB 0002E         CALLS   #1, LIB$GET_EF                       ;
                             53             50  D0 00031         MOVL    R0, STS                              ;
                             2C             53  E9 00034         BLBC    STS, 1$                              ; 2324
                                            7E  7C 00037         CLRQ    -(SP)                                ; 2329
                                            7E  7C 00039         CLRQ    -(SP)                                ;
                             7E      0214   8F  3C 0003B         MOVZWL  #532, -(SP)                          ;
                                       16   A4  9F 00040         PUSHAB  MSG_BUFFER                           ;
                                            7E  7C 00043         CLRQ    -(SP)                                ;
                                       2C   AE  9F 00045         PUSHAB  IOSB                                 ;
                             7E        71   8F  9A 00048         MOVZBL  #113, -(SP)                          ;
                                            64  DD 0004C         PUSHL   CHAN                                 ;
                                       2C   AE  DD 0004E         PUSHL   EVENT_FLAG                           ;
                             66             0C  FB 00051         CALLS   #12, SYS$QIOW                        ;
                             53             50  D0 00054         MOVL    R0, STS                              ;
                                            5E  DD 00057         PUSHL   SP                                   ; 2335
                             67             01  FB 00059         CALLS   #1, LIB$FREE_EF                      ;
                             04             50  E8 0005C         BLBS    STATUS, 1$                           ; 2336
                                            50  DD 0005F         PUSHL   STATUS                               ;
                                            13  11 00061         BRB     3$                                   ;
                             5D             53  E9 00063 1$:     BLBC    STS, 8$                              ; 2338
                             0F        0C   AE  E8 00066         BLBS    QIOW_STS, 4$                         ; 2340
                    0870     8F        0C   AE  B1 0006A         CMPW    QIOW_STS, #2160                      ;
                                            07  13 00070         BEQL    4$                                   ;
                             7E        0C   AE  3C 00072 2$:     MOVZWL  QIOW_STS, -(SP)                      ; 2342
                                          0091  31 00076 3$:     BRW     13$                                  ;
                    0870     8F        0C   AE  B1 00079 4$:     CMPW    QIOW_STS, #2160                      ; 2346
                                            05  13 0007F         BEQL    5$                                   ;
                             50        0E   AE  3C 00081         MOVZWL  TRANSF_BYTES, R0                     ;
                                                04 00085         RET                                          ;
                             11        05   A2  E8 00086 5$:     BLBS    5(R2), 6$                            ; 2351
                                            5B  D5 0008A         TSTL    CCB                                  ; 2357
                                            76  13 0008C         BEQL    12$                                  ;
               71       FC   AB             01  E1 0008E         BBC     #1, -4(CCB), 12$                     ; 2358
                                00000000G   00  16 00093         JSB     BAS$$CB_POP                          ;
                                            69  11 00099         BRB     12$                                  ; 2359
                                       12   A2  B5 0009B 6$:     TSTW    18(R2)                               ; 2365
                                            1B  13 0009E         BEQL    7$                                   ;
 04    AE      00       12   A2 FF676980    8F  7A 000A0         EMUL    #-10000000, 18(R2), #0, DAYTIM       ; 2368
                                            7E  D4 000AB         CLRL    -(SP)                                ; 2370
                                       08   AE  9F 000AD         PUSHAB  DAYTIM                               ;
                                            7E  7C 000B0         CLRQ    -(SP)                                ;
                  00000000G  00             04  FB 000B2         CALLS   #4, SYS$SCHDWK                       ;
                                            38  11 000B9         BRB     9$                                   ;
                                            5E  DD 000BB 7$:     PUSHL   SP                                   ; 2379
                             65             01  FB 000BD         CALLS   #1, LIB$GET_EF                       ;
                             53             50  D0 000C0         MOVL    R0, STS                              ;
                             33             53  E9 000C3 8$:     BLBC    STS, 10$                             ; 2380
                                            7E  7C 000C6         CLRQ    -(SP)                                ; 2383
                                            7E  7C 000C8         CLRQ    -(SP)                                ;
                                            7E  D4 000CA         CLRL    -(SP)                                ;
                                00000000G   00  9F 000CC         PUSHAB  BPA$WAKEUP                           ;
                                            7E  7C 000D2         CLRQ    -(SP)                                ;
                                       2C   AE  9F 000D4         PUSHAB  IOSB                                 ;
```

```
                   7E       0123   8F  3C 000D7           MOVZWL  #291, -(SP)                          ;
                                   64  DD 000DC           PUSHL   CHAN                                 ;
                              2C   AE  DD 000DE           PUSHL   EVENT_FLAG                           ;
                   66              0C  FB 000E1           CALLS   #12, SYS$QIOW                        ;
                   53              50  D0 000E4           MOVL    R0, STS                              ;
                   0F              53  E9 000E7           BLBC    STS, 10$                             ;
                   84         0C   AE  E9 000EA           BLBC    QIOW_STS, 2$                         ; 2387
                                   5E  DD 000EE           PUSHL   SP                                   ; 2389
                   67              01  FB 000F0           CALLS   #1, LIB$FREE_EF                      ;
                   53              50  D0 000F3 9$:       MOVL    R0, STS                              ;
                   04              53  E8 000F6           BLBS    STS, 11$                             ; 2390
                                   53  DD 000F9 10$:      PUSHL   STS                                  ;
                                   0D  11 000FB           BRB     13$                                  ;
          00000000G  00            00  FB 000FD 11$:      CALLS   #0, SYS$HIBER                        ; 2392
                        001A8028   8F  DD 00104 12$:      PUSHL   #1736744                             ; 2397
          00000000G  00            01  FB 0010A 13$:      CALLS   #1, LIB$SIGNAL                       ;
                                       04 00111           RET                                          ; 2398

; Routine Size:  274 bytes,    Routine Base:  _BPA$CODE + 0438
```

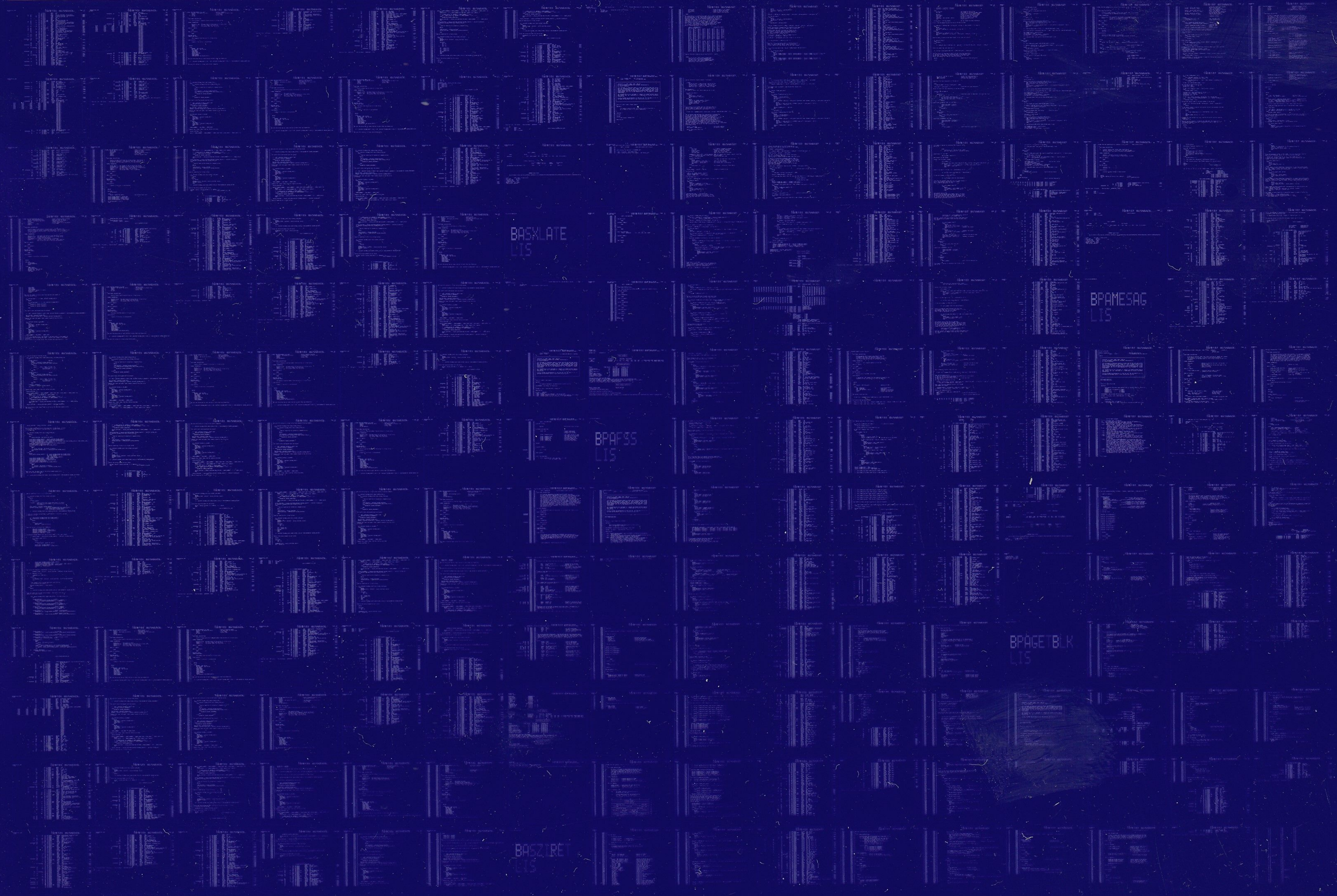
```
1 ROUTINE bpa$mess_rem (firqb) =                        ! Simulates remove receiver
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       The input in firqb is checked and if faulty an error is signalled.
1 !       The receiver is removed by calling, bpa$mess_clear.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       firqb                           =  address to firqb
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       firqb [fqb$b_jobnr]             =   0
1 !       firqb [6, B_] - firqb [31, B_]  =   0
1 !       bpa$a_msg [msg$l_messchan]      =   channel opened to mailbox by
1 !                                           receiver
1 !       bpa$a_msg [msg$t_namedesc]      =   char. string desc. pointing
1 !                                           to logical name of mailbox
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       bpa$a_msg [msg$l_messchan]      =   0 (indicates that the receiver
1 !                                              is removed)
1 !
1 ! ROUTINE VALUE:
1 !
1 !       1  -  success
1 !       ?  -  failure
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2    BEGIN
2
2    MAP
2       firqb : REF $fqb_def;                           ! Defines firqb
2
2 ! Check input.
2
2    IF NOT CH$FAIL (CH$FIND_NOT_CH (26, CH$PTR (firqb [6, b_]), 0))
2    THEN
2       RETURN SIGNAL (badfuo);
2
2    IF .firqb [fqb$b_jobnr] NEQU 0 THEN RETURN SIGNAL (errerr);
2
2 ! Perform 'remove receiver'
2    bpa$mess_clear ();
2    RETURN 1;
1    END;                                               !End of bpa$mess_rem
```

```
                                              0004 00000 BPA$MESS_REM:
                                                               .WORD   Save R2                        ; 2399
                      52         04    AC  D0 00002            MOVL    FIRQB, R2                      ; 2443
     06   A2          1A               00  3B 00006            SKPC    #0, #26, 6(R2)                 ;
                                       02  12 0000B            BNEQ    1$                             ;
                                       51  D4 0000D            CLRL    R1                             ;
                                       51  D5 0000F  1$:       TSTL    R1                             ;
                                       08  13 00011            BEQL    2$                             ;
                           001A8090    8F  DD 00013            PUSHL   #1736848                       ; 2445
                                       0B  11 00019            BRB     3$                             ;
                                 05    A2  95 0001B  2$:       TSTB    5(R2)                          ; 2447
                                       0E  13 0001E            BEQL    4$                             ;
                           001A8210    8F  DD 00020            PUSHL   #1737232                       ;
           00000000G  00               01  FB 00026  3$:       CALLS   #1, LIB$SIGNAL                 ;
                                           04 0002D            RET                                    ;
                0000V CF               00  FB 0002E  4$:       CALLS   #0, BPA$MESS_CLEAR             ; 2450
                      50               01  D0 00033            MOVL    #1, R0                         ; 2451
                                           04 00036            RET                                    ; 2452

; Routine Size:  55 bytes,    Routine Base:  _BPA$CODE + 054A
; 1084          2453   1 ROUTINE exit_handler (                         ! Exit handler
; 1085          2454   1        exit_reason                             ! reason
; 1086          2455   1     ) : NOVALUE =
; 1087          2456   1
; 1088          2457   1 !++
; 1089          2458   1 ! FUNCTIONAL DESCRIPTION:
; 1090          2459   1 !
; 1091          2460   1 !       This is the exit handler for the BASIC-PLUS message handler.
; 1092          2461   1 !       It deletes the system logical name defined by this user, in
; 1093          2462   1 !       case it hasn't been deleted already.
; 1094          2463   1 !
; 1095          2464   1 ! FORMAL PARAMETERS:
; 1096          2465   1 !
; 1097          2466   1 !       EXIT_REASON.rl.r          Not used
; 1098          2467   1 !
; 1099          2468   1 ! IMPLICIT INPUTS:
; 1100          2469   1 !
; 1101          2470   1 !       NONE
; 1102          2471   1 !
; 1103          2472   1 ! IMPLICIT OUTPUTS:
; 1104          2473   1 !
; 1105          2474   1 !
; 1106          2475   1 ! ROUTINE VALUE:
; 1107          2476   1 !
; 1108          2477   1 !       NONE
; 1109          2478   1 !
; 1110          2479   1 ! SIDE EFFECTS:
; 1111          2480   1 !
; 1112          2481   1 !       Deletes the system logical name, if there is one.
; 1113          2482   1 !
; 1114          2483   1 !--
; 1115          2484   1
; 1116          2485   2     BEGIN
; 1117          2486   2     bpa$mess_clear ();
; 1118          2487   2     RETURN;
; 1119          2488   1     END;                                       !End of exit_handler



                                  0000 00000 EXIT_HANDLER:
                                                  .WORD   Save nothing                                    ; 2453
              0000V  CF              00  FB 00002         CALLS   #0, BPA$MESS_CLEAR                              ; 2486
                                         04 00007         RET                                             ; 2488

; Routine Size:  8 bytes,    Routine Base:  _BPA$CODE + 0581


; 1120          2489   1
; 1122        2490  1 GLOBAL ROUTINE bpa$mess_clear =                       ! Removes a permanent receiver
; 1123        2491  1
; 1124        2492  1 !++
; 1125        2493  1 ! FUNCTIONAL DESCRIPTION:
; 1126        2494  1 !
; 1127        2495  1 !       If the caller isn't a declared receiver, success is returned.
; 1128        2496  1 !       The reciver mailbox is deleted, the channel to the mailbox is
; 1129        2497  1 !       deassigned, the mailbox logical name is deleted and the channel is
; 1130        2498  1 !       zeroed to indicate that the caller no longer is a receiver.
; 1131        2499  1 !
; 1132        2500  1 ! FORMAL PARAMETERS:
; 1133        2501  1 !
; 1134        2502  1 !       NONE
; 1135        2503  1 !
; 1136        2504  1 ! IMPLICIT INPUTS:
; 1137        2505  1 !
; 1138        2506  1 !       bpa$a_msg [msg$l_messchan]   =   channel opened to mailbox by receiver
; 1139        2507  1 !       bpa$a_msg [msg$t_namedesc]   =   char. string desc. pointing to the
; 1140        2508  1 !                                        logical name of mailbox
; 1141        2509  1 !
; 1142        2510  1 ! IMPLICIT OUTPUTS:
; 1143        2511  1 !
; 1144        2512  1 !       bpa$a_msg [msg$l_messchan]   =   0
; 1145        2513  1 !
; 1146        2514  1 ! ROUTINE VALUE:
; 1147        2515  1 !
; 1148        2516  1 !       success  -  1
; 1149        2517  1 !       failure  -  ?
; 1150        2518  1 !
; 1151        2519  1 ! SIDE EFFECTS:
; 1152        2520  1 !
; 1153        2521  1 !       NONE
; 1154        2522  1 !
; 1155        2523  1 !--
; 1156        2524  1
; 1157        2525  2     BEGIN
; 1158        2526  2
; 1159        2527  2     LOCAL
; 1160        2528  2         sts;                                          ! Status received from
; 1161        2529  2
; 1162        2530  2                                                       ! calls to sys. serv.
; 1163        2531  2
; 1164        2532  2     BIND
; 1165        2533  2         chan = bpa$a_msg [msg$l_messchan],            ! Channel
; 1166        2534  2         lognam_desc = bpa$a_msg [msg$t_namedesc];            ! Char. string desc.
; 1167        2535  2
; 1168        2536  2                                                       ! of mailbox name
; 1169        2537  2 ! Is the caller a declared receiver? If not, then return success.
; 1170        2538  2
; 1171        2539  2     IF .bpa$a_msg [msg$l_messchan] EQLU 0 THEN RETURN 1;
; 1172        2540  2
; 1173        2541  2 ! Delete the mailbox
; 1174        2542  2
; 1175        2543  2     IF NOT (sts = $delmbx (chan = .chan)) THEN RETURN SIGNAL (.sts);
; 1176        2544  2
; 1177        2545  2 ! Deassign the channel
; 1178        2546  2
; 1179          2547  2        IF NOT (sts = $dassgn (chan = .chan)) THEN RETURN SIGNAL (.sts);
; 1180          2548  2
; 1181          2549  2  ! Delete the logical name
; 1182          2550  2
; 1183          2551  2        IF NOT (sts = $dellog (lognam = lognam_desc)) THEN RETURN SIGNAL (.sts);
; 1184          2552  2
; 1185          2553  2  ! Clear chan to indicate that the receiver is removed.
; 1186          2554  2        chan = 0;
; 1187          2555  2        RETURN 1;
; 1188          2556  1        END;                                   !End of bpa$mess_clear


                                                   CHAN=                     BPA$A_MSG
                                                   LOGNAM_DESC=              BPA$A_MSG+4
                                                             .EXTRN   SYS$DELLOG

                                         000C 00000          .ENTRY   BPA$MESS_CLEAR, Save R2,R3               ; 2490
                       53 00000000'   EF   9E 00002          MOVAB    BPA$A_MSG, R3                            :
                       50             63   D0 00009          MOVL     BPA$A_MSG, R0                            ; 2539
                                      3E   13 0000C          BEQL     3$                                       :
                                      50   DD 0000E          PUSHL    R0                                       ; 2543
            00000000G  00             01   FB 00010          CALLS    #1, SYS$DELMBX                           :
                       52             50   D0 00017          MOVL     R0, STS                                  :
                       23             52   E9 0001A          BLBC     STS, 1$                                  :
                                      63   DD 0001D          PUSHL    CHAN                                     ; 2547
            00000000G  00             01   FB 0001F          CALLS    #1, SYS$DASSGN                           :
                       52             50   D0 00026          MOVL     R0, STS                                  :
                       14             52   E9 00029          BLBC     STS, 1$                                  :
                                      7E   D4 0002C          CLRL     -(SP)                                    ; 2551
                                04    A3   9F 0002E          PUSHAB   LOGNAM_DESC                              :
                                      7E   D4 00031          CLRL     -(SP)                                    :
            00000000G  00             03   FB 00033          CALLS    #3, SYS$DELLOG                           :
                       52             50   D0 0003A          MOVL     R0, STS                                  :
                       0A             52   E8 0003D          BLBS     STS, 2$                                  :
                                      52   DD 00040 1$:      PUSHL    STS                                      :
            00000000G  00             01   FB 00042          CALLS    #1, LIB$SIGNAL                           :
                                           04 00049          RET                                               :
                                      63   D4 0004A 2$:      CLRL     CHAN                                     ; 2554
                       50             01   D0 0004C 3$:      MOVL     #1, R0                                   ; 2555
                                           04 0004F          RET                                               ; 2556

; Routine Size:  80 bytes,    Routine Base:  _BPA$CODE + 0589


; 1189          2557  1
; 1190          2558  1 END                                            !End of module
; 1191          2559  1
; 1192          2560  0 ELUDOM




                                                             .EXTRN   LIB$SIGNAL

;                            PSECT SUMMARY
;
```

```
;       Name                            Bytes                           Attributes
;
;  _BPA$DATA                                608  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;  _BPA$CODE                               1497  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)




;                             Library Statistics
;
;                                              -------- Symbols --------        Pages        Processing
;       File                                   Total    Loaded   Percent        Mapped       Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1            9776        26         0          581          00:01.2





;                                   COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS$:BPAMESAG/OBJ=OBJ$:BPAMESAG MSRC$:BPAMESAG/UPDATE=(ENH$:BPAMESAG)

; Size:           1484 code + 621 data bytes
; Run Time:          00:36.9
; Elapsed Time:      01:26.2
; Lines/CPU Min:      4163
; Lexemes/CPU-Min: 29056
; Memory Used:  239 pages
; Compilation Complete
```

0034 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY


BASXLATE
LIS
BPAMESAG
LIS
BPAFSS
LIS
BPAGETBLK
LIS
BASZIRET
LIS

0035 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

BOOTBLOCK
MAP

STACONFIG
MAP

SYSBOOT
MAP

BPAMOVTUC
LIS

BOOTS

BPASSDEAS
LIS

BOOT58
MAP

STANDCONF
MAP

CONFIGURE
MAP

BPAWAKEUP
LIS

STASYSGEN
MAP

BPASETPRI
LIS